UNIVERSITÀ DI PARMA
Dipartimento di Ingegneria e Architettura

Cryptography: Symmetric Key Establishment

Luca Veltri

(mail.to: luca.veltri@unipr.it)

Course of Cybersecurity, 2022/2023

http://netsec.unipr.it/veltri

Key management

- In a secured communication, users must setup the details of the cryptography
  - **In some cases this may require <u>sharing a symmetric key</u>**
  - **In others it may require <u>obtaining the other party's public key</u>**
- In both cases, an issue is how to securely distribute these keys
  - **in case of symmetric key**
    - must be exchanged over a secure communication channel
      - both confidentiality and data authentication must be guaranteed
  - **in case of public key**
    - keys can be openly exchanged (authenticated) over an insecure communications channel
      - only message authentication must be guaranteed
        - » the exchange is less troublesome
        - » for example, by using digital certificates
- Often secure systems fail due to a break in the key distribution

Long-term vs Short-term Keys

- Cryptographic system and protocols usually deal with two types of keys with two different lifetimes:
  - **short-term keys**
    - also called as session keys
    - used to secure a communication
      - data confidentiality
      - data authentication and integrity
    - used only for a limited interval of time
    - setup through a proper KE protocol
  - **long-term keys**
    - also referred to as long-term secrets
    - can be used for
      - peer authentication
      - securing key exchanges

(Secret) Key establishment

- Secret key establishment is a process or protocol whereby a shared secret becomes available to two or more parties, for subsequent cryptographic use
- Two approaches are possible:
  - **Key pre-distribution**
    - key establishment schemes whereby the resulting established keys are completely determined “a priori” by initial keying material
      - used mainly only for long-term secret keys
  - **Dynamic key establishment (or key exchange)**
    - those schemes whereby the key established by a pair (or group) of users varies on subsequent executions
    - subdivided into (see later):
      - key transport
      - key agreement

Key establishment (cont.)

- Dynamic key establishment may be broadly subdivided into:
  - **key transport**
    - A key transport protocol or mechanism is a key establishment technique where one party creates or otherwise obtains a secret value, and securely transfers it to the other(s)
  - **key agreement**
    - A key agreement protocol or mechanism is a key establishment technique in which a shared secret is derived by two (or more) parties as a function of information contributed by, or associated with, each of these, (ideally) such that no party can predetermine the resulting value
- Additional variations exist, including various forms of key derivation and key update

Use of trusted servers

- Some key establishment protocols involve a centralized or trusted party, for either or both:
  - **initial system setup**
  - **on-line actions (involving real-time participation)**
- This party is referred to by a variety of names depending on the role played
  - **generically called as trusted third party or trusted server**
- E.g.
  - **authentication server**
  - **key distribution center (KDC)**
  - **certification authority (CA)**

Properties of key establishment

- Properties that a KE protocol may satisfy:
  - **Entity authentication**
    - a party in a key establishment protocol is able to determine the true identity of the other(s) which could possibly gain access to the resulting key
  - **Implicit Key authentication**
    - is the property whereby one party is assured that no other party aside from a specifically identified second party (and possibly additional identified trusted parties) may gain access to a particular secret key
      - the other protocol participant is the only other party that could possibly be in possession of the correct established key
  - **Key confirmation**
    - is the property whereby one party is assured that a second (possibly unidentified) party actually has possession of a particular secret key
  - **Explicit key authentication**
    - both implicit key authentication and key confirmation hold
      - an identified party is known to actually possess a specified key
- A KE protocol is usually called "Authenticated Key Exchange" (AKE) if both entity authentication and explicit key authentication are provided
- In addition:
  - **Key freshness assurance**
    - assurance that the exchanged key is new

Forward secrecy and known-key attacks

- It is important to consider the potential impact of compromise of various types of keying material
  - **even if such compromise is not normally expected**
- In particular, the effect of the following is often considered:
  - **compromise of past session (established) keys**
  - **compromise of long-term secret (symmetric or asymmetric) keys**
- A protocol is said to be "resistant to a known-key attack" if compromise of past session keys doesn’t allow an adversary to compromise future session key exchanges
- A protocol is said to have "(perfect) forward secrecy" if compromise of long-term keys does not compromise past session keys
  - **example: Diffie-Hellman key agreement, in some cases, may provide forward secrecy**

Key transport and derivation using symmetric cryptography

Point-to-point key transport and key derivation

- Key transport or derivation based on a long-term symmetric key K_{ab} shared “a priori” by two parties A and B
  - **the long-term key must be initially distributed over a secure channel or resulting from a key pre-distribution mechanism**
  - **the long-term key is used to establish new session keys K_S**
    - KE protocol (transport/derivation): $K_{AB} \rightarrow K_S$

- Possible techniques:
  - **key transport based on symmetric encryption**
  - **key derivation based on non-reversible functions**
- Variants:
  - **key transport with one pass**
  - **key transport with challenge-response**

Key transport

- Key transport with one pass:

$$A \rightarrow B : \quad E_{Kab}(K_S)$$

  - **both A and B obtain implicit key authentication**
  - **susceptible to known-key attacks through replay attack**

- Additional optional fields might be transferred in the encrypted portion:

$$A \rightarrow B : \quad E_{Kab}(K_S, t_A^{*}, B^{*})$$

  - **field containing redundancy provides explicit key authentication to B and facilitates message modification detection**
  - **timestamp (or sequence number) provides a sort of freshness guarantee to B**
    - avoids replay attacks
  - **a destination identifier prevents message replay back on A**
    - if a timestamp is present it provides also entity authentication to B

Key transport (cont.)

- Key transport with challenge-response:
  - **If anti-replay, explicit key authentication, and entity authentication are desired for B but reliance on timestamps added by A is not, a random value or sequence number n_B may be used**
  - **the cost is an additional message**

$$A \leftarrow B: \quad n_B$$

$$A \rightarrow B: \quad E_{Kab}(K_S, n_B, B^*)$$

- If it is required that the session key K_S be a function of inputs from both parties and <u>mutual authentication</u>:

$$A \leftarrow B: \quad n_B$$

$$A \rightarrow B: \quad E_{Kab}(k_1, n_A, n_B, B^*)$$

$$A \leftarrow B: \quad E_{Kab}(k_2, n_B, n_A, A^*)$$

  - **$K_S = f(k_1, k_2)$**

Key transport (cont.)

- Vulnerabilities:
  - **Any previous exchange does not offer forward secrecy**
    - they fail if the long-term key K_{ab} is compromised
    - for this reason they may be inappropriate for many applications
- Note:
  - **Authentication protocols which employ encryption, including the above key exchange schemes, may require that the encryption function has a built-in data integrity mechanism to detect message modification**
    - i.e. Authenticated Encryption (AE)

Key derivation

- Key exchange may be achieved also by dynamic key derivation where the derived session key is based on per-session random input provided by one party

$$A \rightarrow B : \quad r_A$$

  - **single message**
  - **the session key is computed as $K_S = f_{Kab}(r_A)$**
    - where f is a cryptographic function, like $E_K()$ or $MAC_K()$
  - **provides to both A and B implicit key authentication**
  - **susceptible to known-key attacks through replay attack**
- The random number r_A here may be replaced by other time-variant parameters
  - **e.g. a timestamp t_A, provides an implicit key freshness property**
    - avoids replay attack to B
    - avoids A can force a given key X

Key derivation (cont.)

- In general is preferred a keyed one-way function (like $MAC_K()$) for $f_K()$, in place of an encryption function $E_K()$
  - **A cannot control the value of K_S**

- The simple key derivation scheme can be further extended with two or three passes in order to provide:
  - **contribution of B to the creation of K_S**
  - **mutual entity authentication**

Example of authenticated key derivation protocol

- Authenticated Key Exchange Protocol AKEP2
  - **provides mutual entity authentication, key freshness guarantee, and implicit key authentication**
  - **A and B exchange 3 messages**
- *Setup*:
  - **A and B share long-term symmetric keys K_{ab}**

$$A \rightarrow B: \quad r_A$$
$$A \leftarrow B: \quad X_B = \{B, A, r_A, r_B\}\ ,\ MAC_{Kab}(X_B)$$
$$A \rightarrow B: \quad X_A = \{A, r_B\}\ ,\ MAC_{Kab}(X_A)$$

  - **$K_S = f_{K'}(r_B)$**
    - where $f_K()$ is a keyed one-way function or encryption function
    - K' is the key used to derive Ks, usually obtained from K_{ab}

Key transport
using asymmetric cryptography

Key transport based on public-key encryption

- One party may choose a symmetric key and transfer it to a second, using that party's encryption public key

$$A \rightarrow B: \ \{K_S\}KU_B$$

  - **this provides (implicit) key authentication to the originator (A)**
    - only the intended recipient has the private key allowing decryption
    - the originator (A) obtains neither entity authentication nor key confirmation
  - **the second party (B) has no assurances regarding the source of the key and the timeliness**
    - even if A sends to B: $\{K_S, A, T_A\}KU_B$, since KU_B is public
  - **such additional assurances may be obtained through use of further techniques including:**
    - additional messages, with public-key encryption
      - using challenge-response like key transport based on symmetric key
    - digital signature

Key transport using public-key encryption and signature

- Encryption and signature primitives may be used to provide respectively:
  - **privacy of keying material**
  - **source authentication**
- Some possible approaches:
  - **sign the key and separately public-key encrypt the (unsigned) key**
  - **sign the key, then public-key encrypt the signed key**
  - **public-key encrypt the key, then sign the encrypted key**

Key transport using public-key encryption and signature in one-pass

- Encrypting and signing separately:
  - **An option is to sign the key and encrypt the key:**

$$A \rightarrow B: \quad \{A, K_s, t_A^*\}KU_B, Sign_A(B, K_s, t_A^*)$$

- Encrypting signed keys:
  - **A variation is to encrypt signed blocks:**

$$A \rightarrow B: \quad \{A, K_s, t_A^*, Sign_A(B, K_s, t_A^*)\}KU_B$$

- Signing encrypted keys:
  - **In contrast to encrypting signed keys, one may sign encrypted keys**

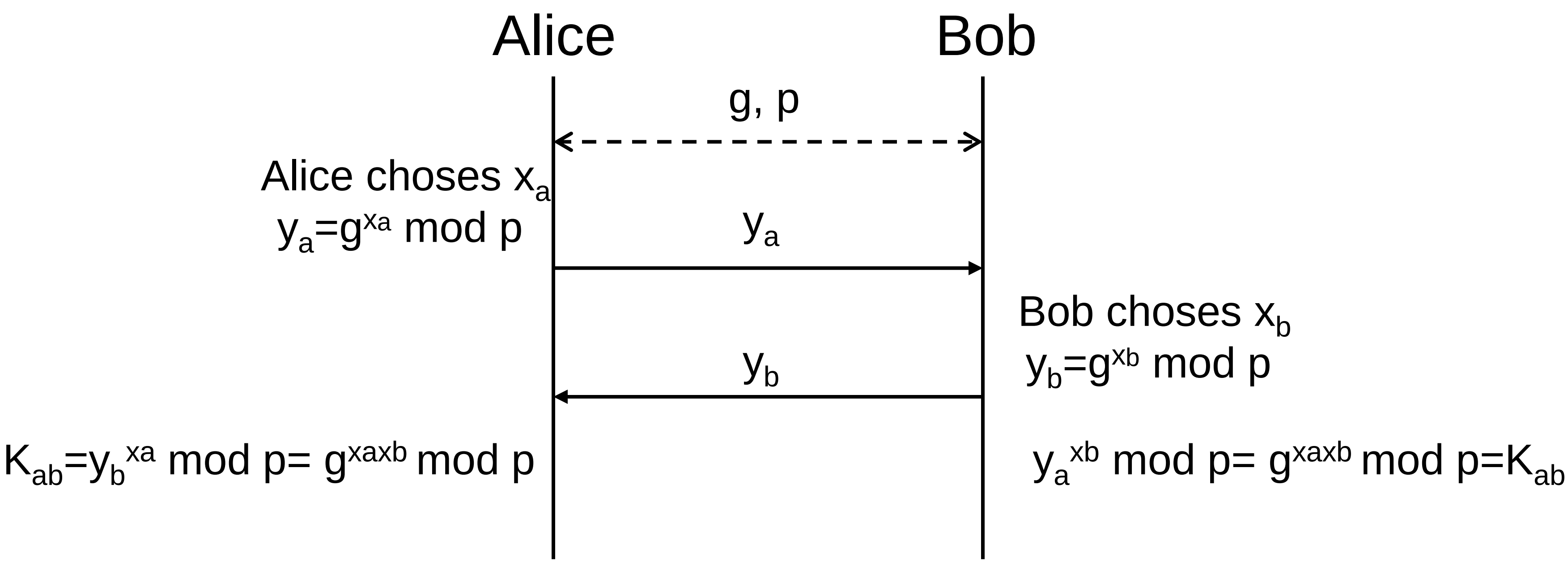
$$A \rightarrow B: \quad t_A^*, Q, Sign_A(B, t_A^*, Q)$$

where $Q=\{A, K_s\}KU_B$

Key agreement
using asymmetric cryptography

Diffie-Hellman key exchange

- The first publicly known public-key agreement protocol was the Diffie-Hellman exponential key exchange



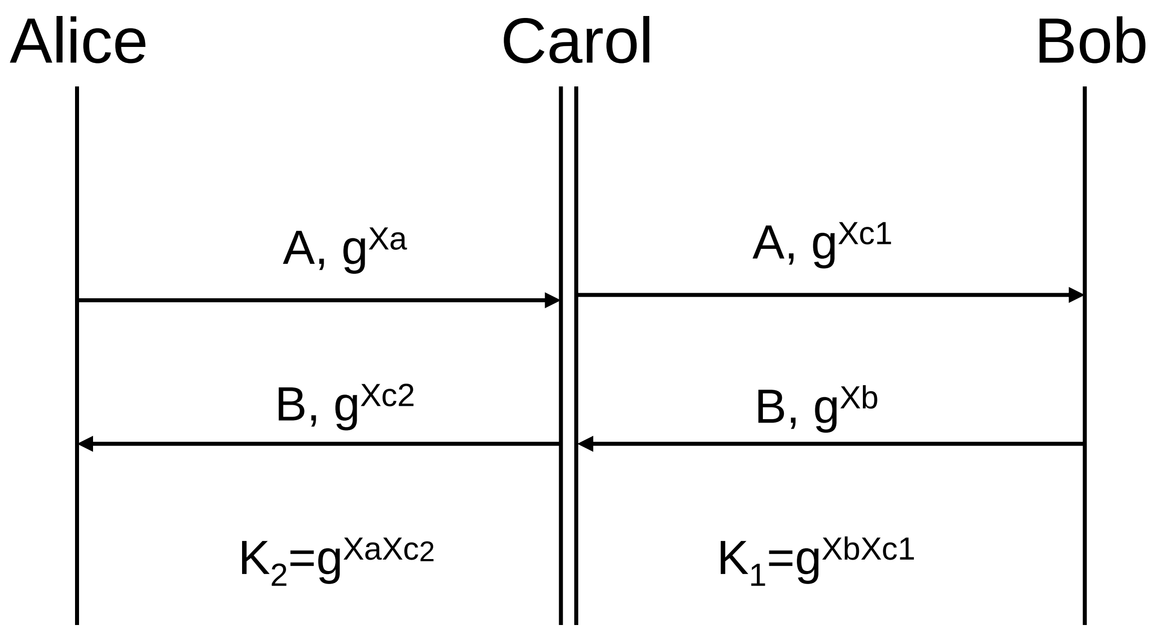
DH modes

- Different ways of using DH:
  - **Fixed Diffie-Hellman**
    - each party already has the public part (DH public key) of the other entity
  - **Anonymous Diffie-Hellman**
    - DH exchange
    - susceptible to Man-in-the-Middle attacks
  - **Ephemeral Diffie-Hellman (EDH)**
    - authenticated DH exchange

- Elliptic Curve variant:
  - **Elliptic Curve DH (ECDH) uses elliptic curves instead of the multiplicative group of integers modulo p**

Diffie-Hellman Vulnerability (MITM attack)

- Anonymous Diffie-Hellman key exchange, does not provide authentication of the parties, and is thus vulnerable to Man-in-the-middle attacks
  - **a third party C may run two separate exchanges with A and B, convincing the two peers that the exchange ended successfully between them**

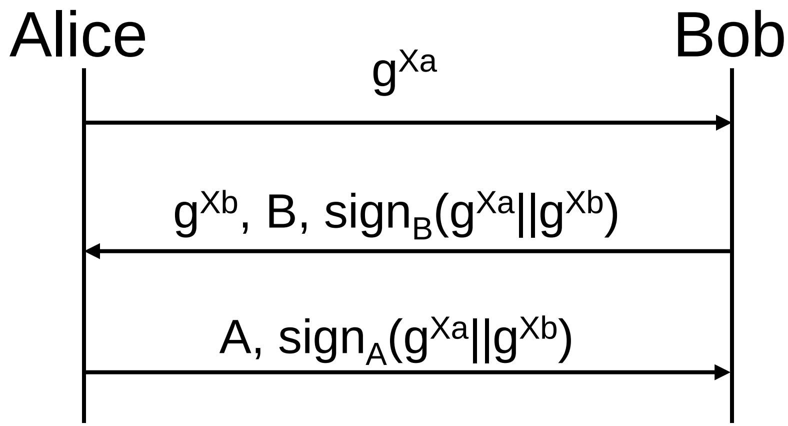


Authenticated DH Exchange

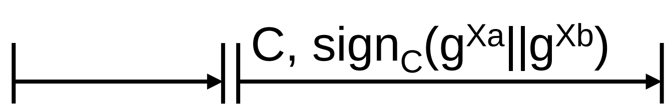
- A wide variety of schemes and protocols have been developed to provide authenticated DH key agreement to prevent man-in-the-middle and related attacks
- These methods generally use:
  - **Public/private key pairs**
  - **Shared secrets**

Authenticated DH Exchange (cont.)

- The simplest way to authenticate the DH exchange could be to sign the DH exponentials



➢ **It does not guarantee implicit key authentication nor confirmation**
  - the authenticated entity may be different from the peer that exchanged the secret key
    - a third party C can convince B that the exchange was run with C, by simply replacing the third message with:



    - this attack does not result in a breach of secrecy of the key

Authenticated DH Exchange (cont.)

- Solution: uses signature and Enc
  - **variant of Station-to-Station (STS) protocol**

$$A \rightarrow B: \quad g^{Xa}$$
$$A \leftarrow B: \quad g^{Xb}, E_{Ks}(B \,||\, Sign_B(g^{Xa} \,||\, g^{Xb}))$$
$$A \rightarrow B: \quad E_{Ks}(A \,||\, Sign_A(g^{Xa} \,||\, g^{Xb}))$$

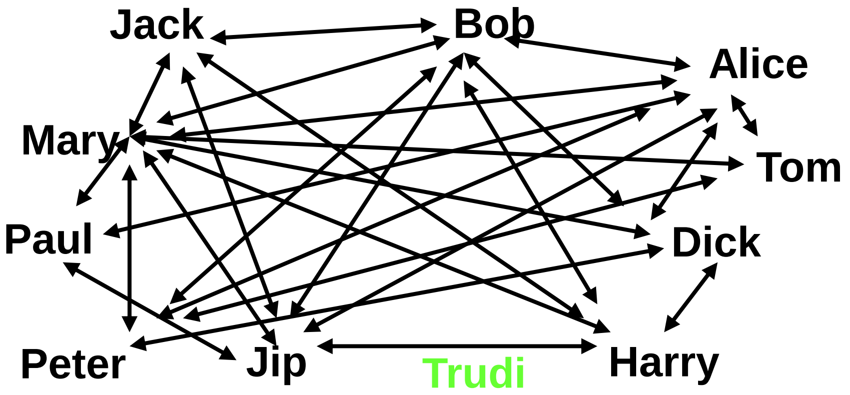
- Solution: uses signature and MAC
  - **SIGMA Protocol:**
    - signatures authenticate the DH exponentials
    - MACs bind the key to identities
      - MAC is performed with a key K_m derived by $K_S=g^{XaXb}$

$$A \rightarrow B: \quad g^{Xa}$$
$$A \leftarrow B: \quad B, g^{Xb}, Sign_B(g^{Xa} \,||\, g^{Xb}), MAC_{Km}(B)$$
$$A \rightarrow B: \quad A, Sign_A(g^{Xa} \,||\, g^{Xb}), MAC_{Km}(A)$$

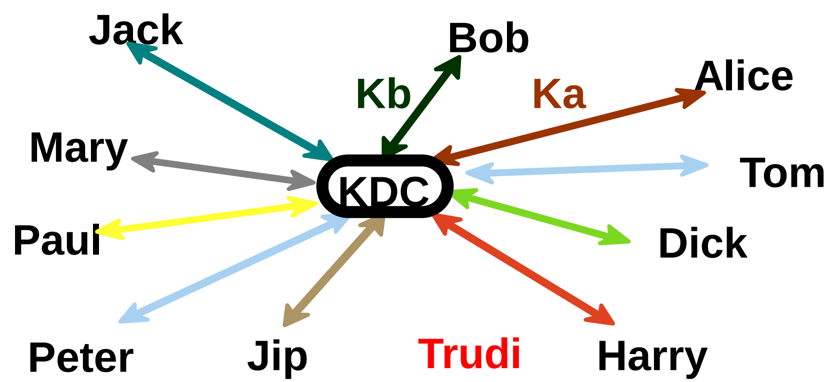
Server-based Key Distribution

Direct trust vs. Trusted intermediaries

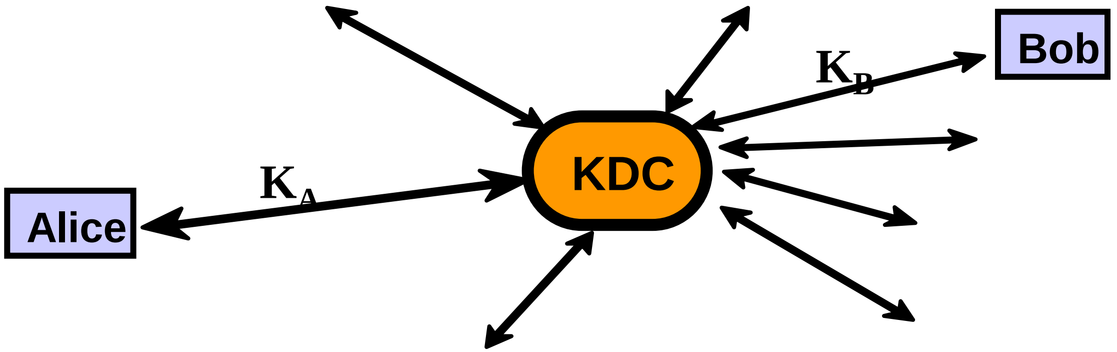
- With *N* nodes, each node must authenticate each other.. *N*-1 keys maintained by each node

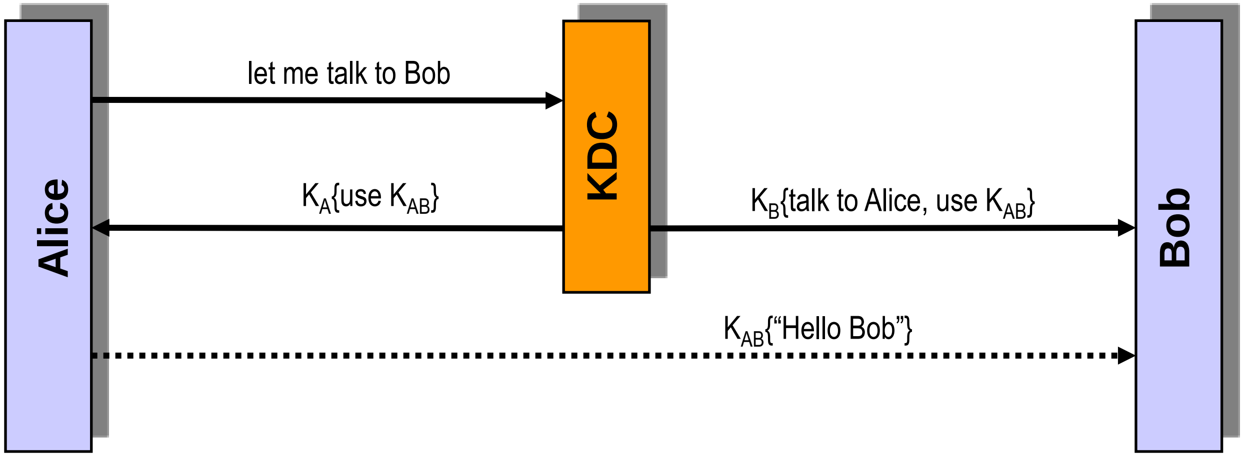


- Possible solution: Trusted intermediate party
  - **Key Distribution Center (KDC) or Key Translation Center (KTC)**
  - **similar to CAs for public-key cryptography**

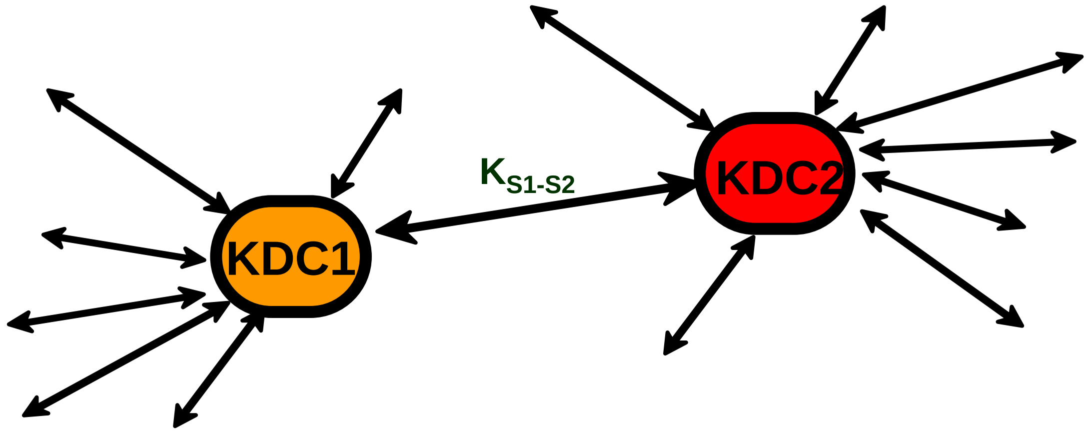


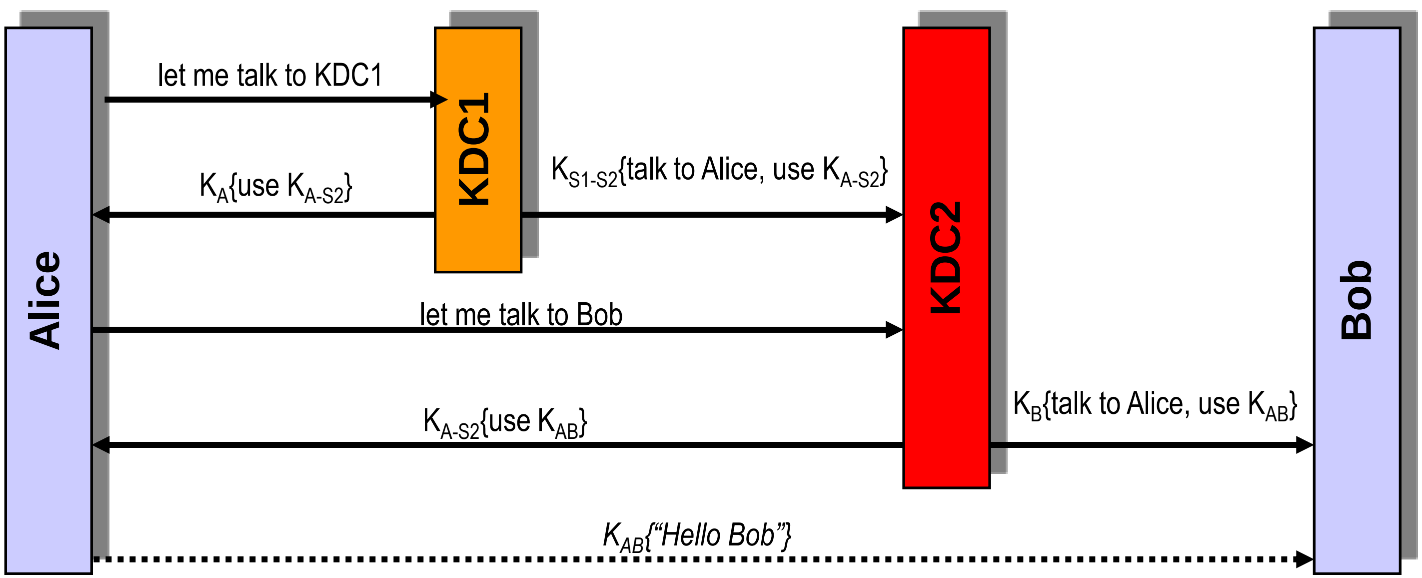
Key distribution with a trusted intermediary





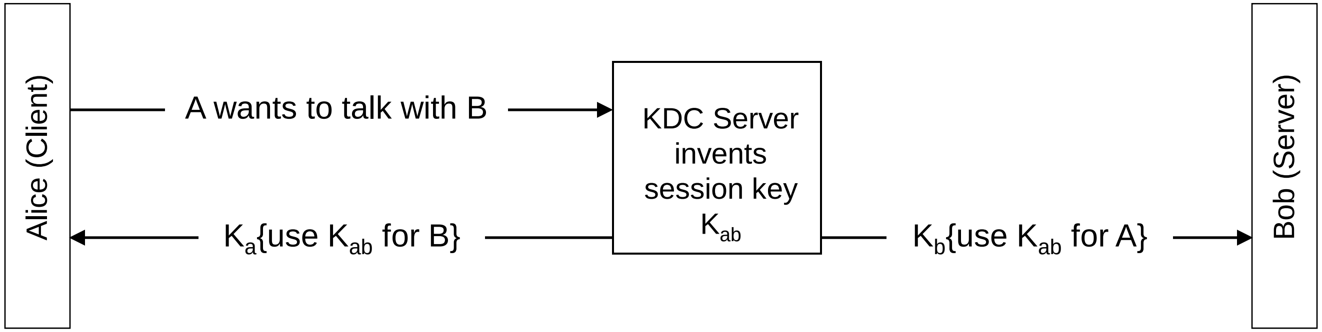
Key distribution with multiple trusted intermediaries





Key Distribution in practice

- Key Distribution in theory (Client-Server)



- Key Distribution in practice (Client-Server)
  - **e.g. used by Kerberos protocol**